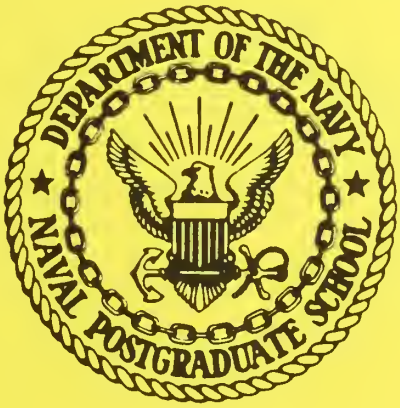
NPS-62-86-007

NAVAL POSTGRADUATE SCHOOL

Monterey, California



SIMILARITY COUNTING ARCHITECTURE
FOR OBJECT DETECTION

CHIN-HWA LEE

JUNE 1986

Annual Report for Period October 1985-September 1986

Prepared for:
al Postgraduate School
terey, CA 93943-5100

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Rear Admiral R. H. Shumaker
Superintendent

D. A. Schrady
Provost

The work reported herein was supported by Naval Electronic Systems Command.

This report was prepared by:

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: NPS-62-86-007	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): Similarity Counting Architecture for Object Detection	5. TYPE OF REPORT & PERIOD COVERED: Annual Report for Period October 1985-September 1986
6. PERFORMING ORG. REPORT NUMBER	
7. AUTHOR(s): Chin-Hwa Lee	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Postgraduate School, Monterey, CA 93943-5100	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: Naval Postgraduate School, Monterey, CA 93943-5100	12. REPORT DATE: June 86
	13. NUMBER OF PAGES: 24
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Parallel Architecture, Content Addressable Memory, Object Detection, Image Correlation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A new algorithm to detect object in image is presented here. It can achieve similar results as the two dimensional correlation method with shorter execution time. An architecture using content addressable memory is implemented in the SCALD CAD environment. The design of the shift counter unit is described in detail.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

S/N 0102-LF-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Similarity Counting Architecture For Object Detection

Chin-Hwa Lee

Electrical & Computer Engineering

Naval Postgraduate School

Monterey, CA 93943

INTRODUCTION

In digital image processing it is often necessary to examine an image and ask the question whether a known object exists in the input image. This is generally referred to as object detection problem. The object can be anywhere in the field of view, and can have different aspect angles or sizes. The method to find the object in the image should be position invariant, scale invariant, and orientation invariant to the difference between it and the model image called template. In aerial photography, satellite IR imagery, or multispectral images such as those collected from LANDSAT, the object in an image may have position shifts, scale change, and aspect change due to different camera position or sensor distortion. For calibrated measuration it is also necessary to associate the views in two different images as belonging to the same object. This is usually called image registration problem. The usual approach for automatic image registration is to do two dimensional correlation or use sequential Similarity Detection method [1,2]. A new algorithm to solve the problems mentioned above is proposed here. First, the algorithm is introduced. This algorithm, can be implemented in a special architecture which uses the associative memory or content addressable memory (CAM). Finally, a logic design of a one-dimensional implementation is discussed here in details.

Similarity Counting Algorithm:

For simplicity let's start the discussion of a one dimensional problem. The input signal is a sequence $f(x)$, where $x=0,1, \ldots, L-1$. The template signal is a sequence of $g(x)$, where $x=0,1, \ldots, M-1$. Assume that there is no need to worry about the aperiodic correlation. A cyclic correlation of $f(x)$ with $g(x)$ is good enough to indicate the existence of the object in the input.

$$f(x) = f(x), \quad x=0,1, \ldots, L-1 \qquad (1)$$

construct a periodic $g_e(x)=g_e(x+nL)$; where n is an integer.

$$g_e(x) = \begin{cases} g(x), & x=0,1, \ldots, M-1 \\ 0, & x=M, \ldots, L-1 \end{cases} \qquad (2)$$

the periodic correlation is:

$$h(x) = \sum_{m=0}^{L-1} f(m)g_e(m+x) \qquad (3)$$

Step 1: Shift operation

input sequence: f(0), f(1), f(2), ..., f(M), ...f(L-1)

template sequence

shifted by one position g(0), g(1), ... g(M-1)

Step 2: Operate and accumulate

f(0), f(1), f(2) .... f(M), ...f(L-1)

g(0), g(1) .... g(M-1)

sum: = P[f(1),g(0)]+P[f(2),g(1)]+...+P[f(M),g(M-1)]

where; P is a multiply operator

Step 3: Assign the sum to output sequence h(x)

h(1):= sum

In sequential algorithms the above steps are performed for all the samples in the output sequence. Therefore, there are M multiply-add operations in the steps for each output point. A total of M*L multiply-add operations is required to get the correlation sequence. Basically, it is an expansive operation.

If this formulation is extended to a two dimensional input f(x.y) of L*L pixels and a two dimensional template g(x,y) of M*M pixels, the shift operation is shown in Fig. 1.

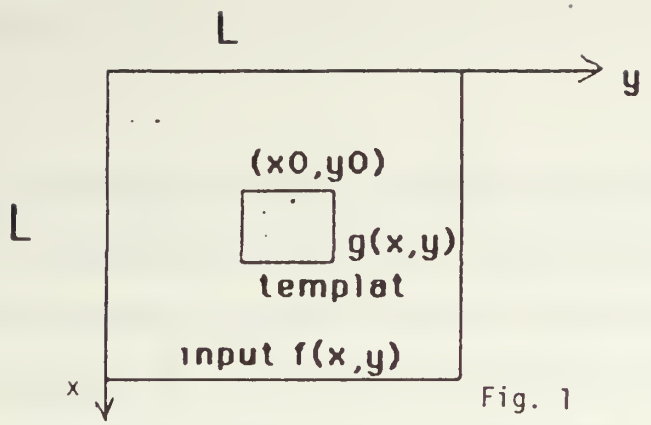


Fig. 1

The "operate and accumulate " will be applied to all the point pairs in the overlapped area between f(x,y) and g(x,y). The accumulated sum is assigned to the output sequence h(x,y). As is obvious, the the total operation to obtain h(x,y) is L^2*M^2 multiply-add operations. For an input image of 400x400 pixels and a template image of 56x52 pixels, the correlation required a typical execution time of two to three hours (10,000 sec) approximately on the VAX11/750. At the end, one examines all possible local maximum points in the correlation image plane. Those are the places where the image signal is most similar to the template image. Barnea recognized this computation difficulty and proposed a different operation in step 2 which is less expansive than the multiplication. The operation in step 2 of the "sequential similarity detection method" is the absolute difference operation, i.e:

$$P[f(x),g(x+n)]= \mid f(x)-g(x+n) \mid \qquad (4)$$

Instead of performing multiplication doing absolute difference saves some time. If the image pixel f(x) at x is similar to the template pixel g(x+n) at x+n the cummulated absolute difference will be small. The original approach of finding the local maximums in the correlation function becomes finding the local minimums in the output image plane. Barnea also recognized that if at a certain position the input image is very different from the template image, the cummulated sum grows quickly. If it exceeds a certain threshold, it is advantageous to abort the "operate and accummulate" step early. Onoe improved the algorithm to allow Automatic threshold settings so that the total number of operations performed was further reduced [2].

A new approach called Similarity counting algorithm is proposed here which can allow algorithm implementation in parallel hardware. However, the new algorithm can also be implemented in sequential software similar to the similarity detection method mentioned above. The operation in step 2 can be generalized to a comparison operation. There can be the exact comparison or the comparision with tolerance. The exact comparison operator P_o can be defined as:

$$P_o[f(x),g(x+n)] = \begin{cases} 1 & , \text{ if } f(x)=g(x+n) \\ 0 & , \text{ otherwise} \end{cases} \quad (5)$$

The comparison operator P_t with tolerance t is then:

$$P_t[f(x),g(x+n)] = \begin{cases} 1 & , |f(x)-g(x+n)| = t \\ 0 & , \text{ otherwise} \end{cases} \quad (6)$$

In this approach the operation has been further reduced to a comparison operator that yields binary results. The sum operation can be replaced by counting the comparison results. Similar to the correlation method, the answer is found to be the local maximums in the output image plane, which is the opposite situation from the sequential similarity detection method. This approach avoids any algebraic calculations and requires only logical comparisons. If it is implemented in sequential software, it can also have all the advantages proposed in Barnea's and Onoe's techniques. The approach of similarity counting with tolerance should have additional advantages whichmakes the technique more orientation independent and scale independent.

From the test results of the simulated algorithms in software, evidence shows those advantages. The new approach also allows better registration of images and the template with different gray tone biases. The results showed that if there is prior knowledge or estimate about the tone bias or geometric dictortion, it's possible to select an appropriate tolerance to minimize the miss registration in the result. Therefore, the new technique in many respects is compar able or superior to both the correlation method and the sequential similarity detection method.

Parallel Hardware Architecture:

A parallel hardware architecture is studied here to implement the similarity counting algorithm. In order to explain the ideas of using parallel hardware such as the associative memory, content addressable memory (CAM), it is necessary to explain an alternative parallel algorithm. For simplicity again, assume a one-dimensional case of f(x) with 8 points and g(x) with 3 points. Let's list the calculated output h(x) for all the points;

$$h(0)= P[f(0),g_e(0)]+P[f(1),g_e(1)]+P[f(2),g_e(2)]$$

$$h(1)= P[f(7),g_e(0)]+P[f(0),g_e(1)]+P[f(1),g_e(2)]$$

$$h(2)= P[f(6),g_e(0)]+P[f(7),g_e(1)]+P[f(0),g_e(2)]$$

$$h(3)= P[f(5),g_e(0)]+P[f(6),g_e(1)]+P[f(7),g_e(2)]$$

$$h(4)= P[f(4),g_e(0)]+P[f(5),g_e(1)]+P[f(6),g_e(2)]$$

$$h(5)= P[f(3),g_e(0)]+P[f(4),g_e(1)]+P[f(5),g_e(2)]$$

$$h(6)= P[f(2),g_e(0)]+P[f(3),g_e(1)]+P[f(4),g_e(2)]$$

$$h(7)= P[f(1),g_e(0)]+P[f(2),g_e(1)]+P[f(3),g_e(2)] \qquad (7)$$

As was mentioned above for similarity counting algorithm P operator for exact match is shown in equation (5) and that for tolerance match is shown in equation (6).

Let's assume that there is a content addressable memory whose functional block diagram is shown in Fig. 2. The content addressable memory has 32 memory cells. Each cell stores an 8-bit byte which is accessible by its address in the same way as the regular random addressable memory (RAM). There is a register called comparant register that can store a desired bit pattern, and the CAM can search that pattern in the cells. It is possible to search part of the bit pattern of the comparant using the bit enable mark. If the bit in the mark is set to one, there will be a match of the corresponding bit between the comparant and the cells. For example,

the comparant is: 100001XX

and the bit enable mask is: 11111100

Bit 2 to bit 7 of all the cells will be matched with those of the comparent. The bit 0 and bit 1 of the comparant will not affect the result because they are masked out in the comparison. It is possible to implement the tolerance match equation (6) of the similarity counting algorithm using the bit enable mask capability of the CAM. For the above arrangement, a byte 10000100 located at address 3 will be found in the CAM operation as shown below.

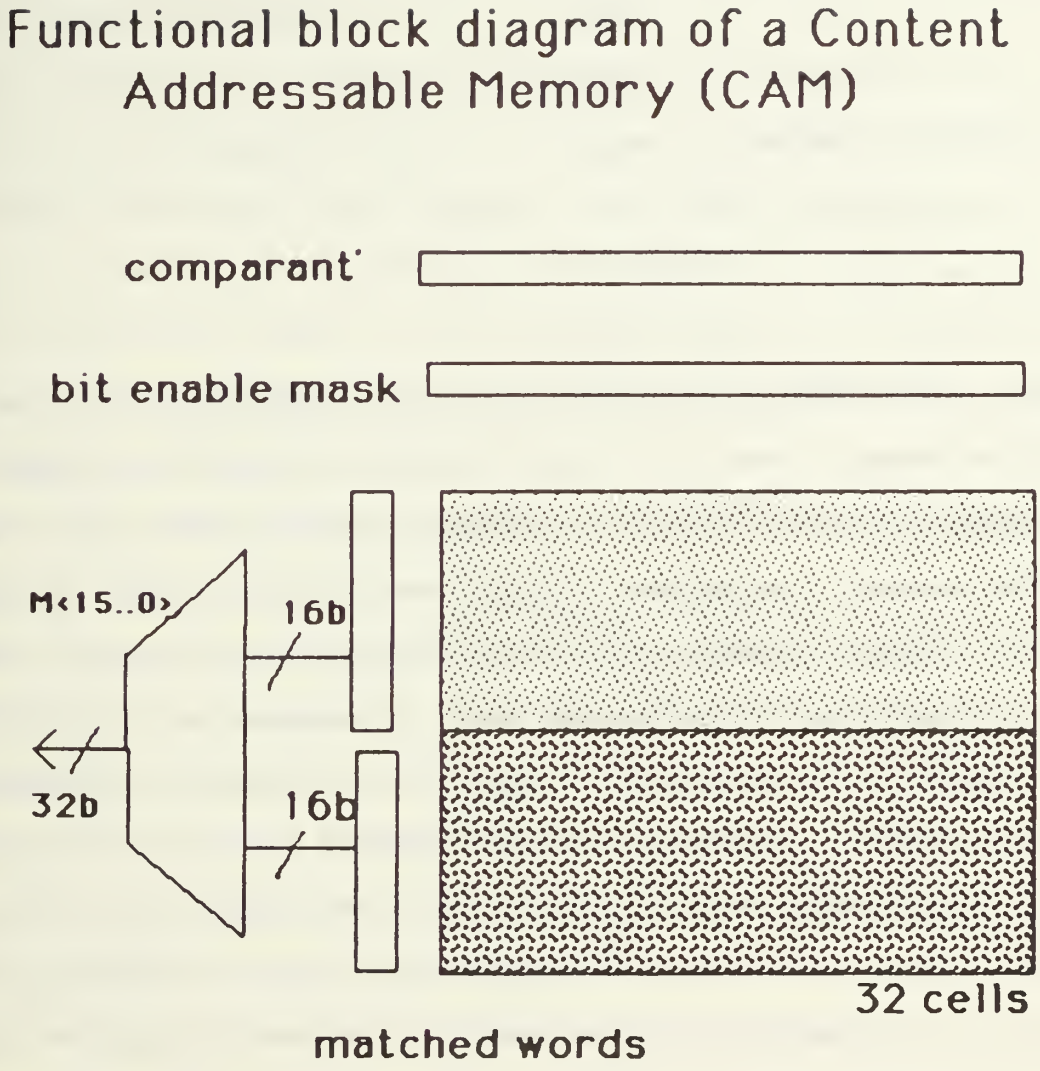


Fig. 2. Functional block diagram of a Content Addressable Memory.

	CAM Content		Address
	.		Cell 0
	.		Cell 1
	.		Cell 2
(found)	10000100	-	cell 3
	.		.
	.		.
	.		.

Assume further that no other cell matches with the comparant for the specified bits in the upper bank of the cells. A match word whose bit pattern corresponding to the cells in the CAM can be read out at the end. Its address can be decoded. For the above example, the read out word will be hexadecimal 1000. If there is another byte 10000101 located at cell address 1, then the read out match word is hexadecimal 5000. By checking the bits in the read out match word, the location of all matches in the CAM cells can be determined.

The CAM can perform the P operation defined in (5) and (6) parallel in space over a number of data stored in the cells. Examine the equations listed in (7). In the usual sequential algorithm the h(x) is calculated one row at a time. It is conceived that in a parallel algorithm the input f(x) can be stored into the cells of the CAM. Each time we take one sample of the template g(x) and use it as the comparant. Then, at one operation time of the CAM, the match word of the CAM yields one column of the equations in (7). The critical issue is about how to use the match word output from the CAM to yield the result. At this point you can see that you need to operate CAM a number of times proportional to the size of the template, M. If everything else can

be done in hardware, the total number of operations for a one dimensional signal is reduced from L*M to M as compared to that of traditional correlation method. This situation is even better than the FFT algorithm which reduce a N^2 problem to a Nlog N problem.

Since the output of a comparison operator P is either binary 0 or 1, the accumulation of the comparison result can be simplified to binary counting. It is decided that parallel binary counters can be used to accumulate the results. If we perform the comparison of the template g(x) in the reverse order, the accumulation starts with the right most column in (7) toward left. This way the index argument of f(x) in P operators decrease its value. If the f(x) is stored in CAM cells in fixed positions, the result in the accumulation has to be shifted toward the least significant direction before it can be added with the later result. In other words, the match word read out need to be shifted so that the accumulations in counters is done in the same way as that described in the rows of (7). It is found that it is advantageous to shift the counter instead of the match word. That can minimize the total number of shifts required. Shown in Fig. 3 is a simplified view of the shift count operation discussed above. A bank of binary counters are connected into a parallel load ring. The bit 0 of the match word enables the counter 0, and bit 1 for the counter 2, and so on. When this operation finishes, the counter has the result of h(0) left in counter 0 which happens to be the index of the first argument of the P operators perfornly most recently. From equations in (7) it is noticed that the index of the f(x), the first argument of P, is the L's complement of the output index of h(x). Therefore, there is a simple last data shuffling necessary.

The parallel algorithm can be summarized as follows;

(1) load the input f(x) into the CAM cells in the forward order.

f(0) → cell 0

f(1) → cell 1

.

.

f(31) → cell 31

(2) load in one comparant from g(x) in the reverse order.

1st time: g(M) → comparant

2nd time: g(M-1)→comparant

.

.

.

Mth time g(0) → comparant

(3) For each of the comparant loads, there is a count shift operation.

(4) At the end, do a data reshuffling based on the L's complement of thecurrent position.

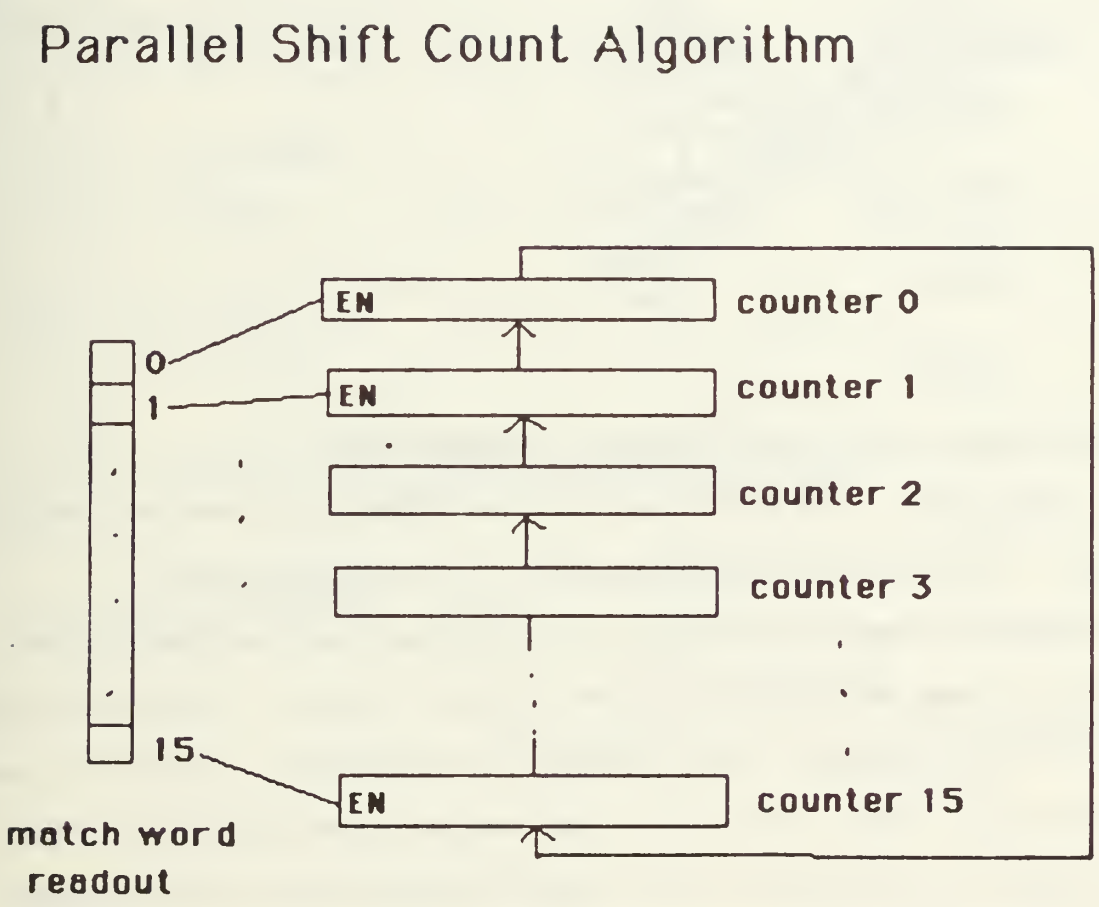
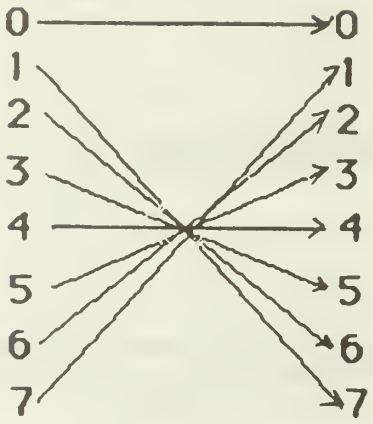


Fig. 3. the shift count algorithm.

If L=8,

Data in	output
Counter	h(x)



A Logic Design for a One-Dimensional Implementation:

Assume that a CAM memory like that described in the previous section exists. Details of the CAM memory will not be discussed here. A one-dimensional implementation of the similarity counting algorithm as a special purpose peripheral subsystem is shown in Fig. 4. There are 4 main parts in this subsystem. The first part is the CAM which stores the input f(x) in cells. The second part is the template where the g(x) is stored. The third part is the shiftcount unit, and the Fourth part is the controller. The subsystem has its usual interface to the host buses. There are decoding circuitry and parallel ports which accepts command from the host and receive f(x) and g(x) to detect the object in the signal. The status of the operation is reported back to the host. If the operation is successful, the result will also be transferred back to the host.

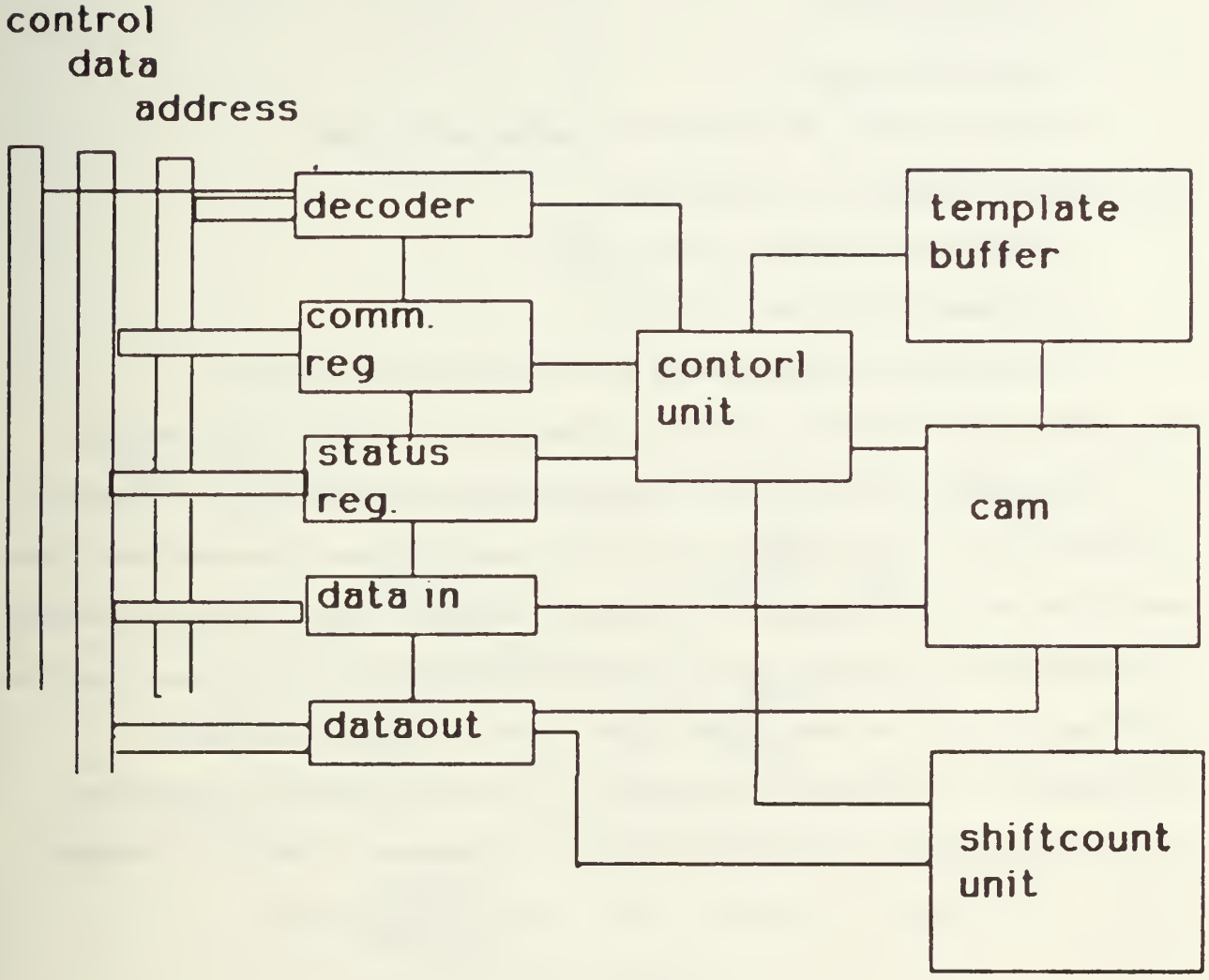


Fig. 4. the special peripheral subsystem.

The system level interface between the host and the subsystem is as follows. The subsystem accepts inputs as;

.Image data, f(x).

.Template data, g(x)

.Match tolerance, to set the bit enable mask in CAM.

The subsystem produces the outputs as;

.Locations of the best match clusters

.The number of the match clusters

.The locations and the counts of the other match clusters

The host should be able to use this information in an application program.

The template buffer is just a regular buffer RAM memory with no special circuitry in it. The details of the shift count unit are discussed here. The operations of the CAM memory, the template buffer, and the shiftcount unit are coordinated by the control unit. To achieve high speed operation and flexible configuration for two dimensional signal cases the controller can be implemented in bit-slice microprocessors. For a one dimensional case the situation is simple, the controller unit is implemented directly in hardware logic. The design of the control unit is not discussed here.

Shift Count Unit:

Following the algorithm discussed in the previous section, the binary counter bank is designed first. Fig. 5 shows the logic diagram of the basic counter called cell 1. Two F163's are used to provide an 8-bit binary counter. The propagation delay in the output from the clock is about 12 nanosecond.

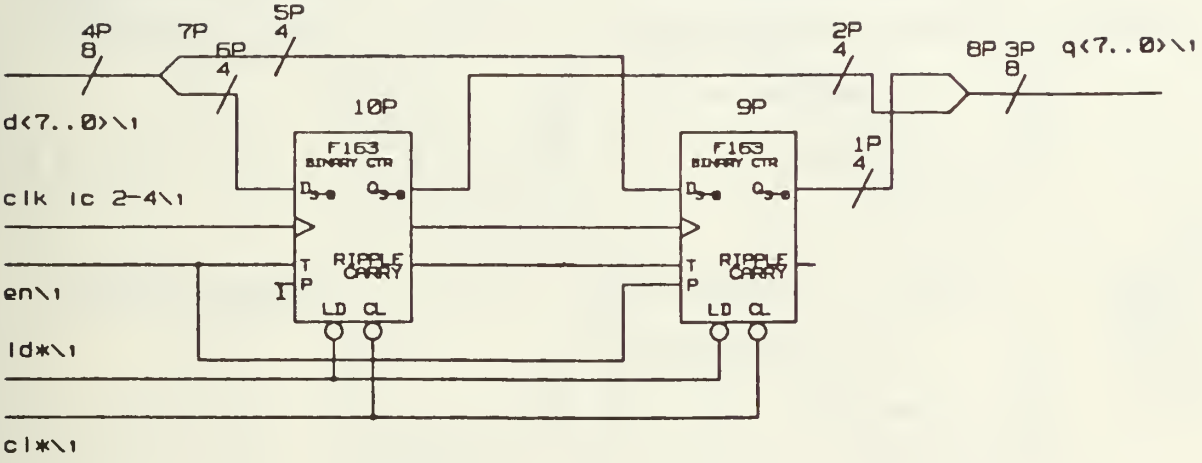
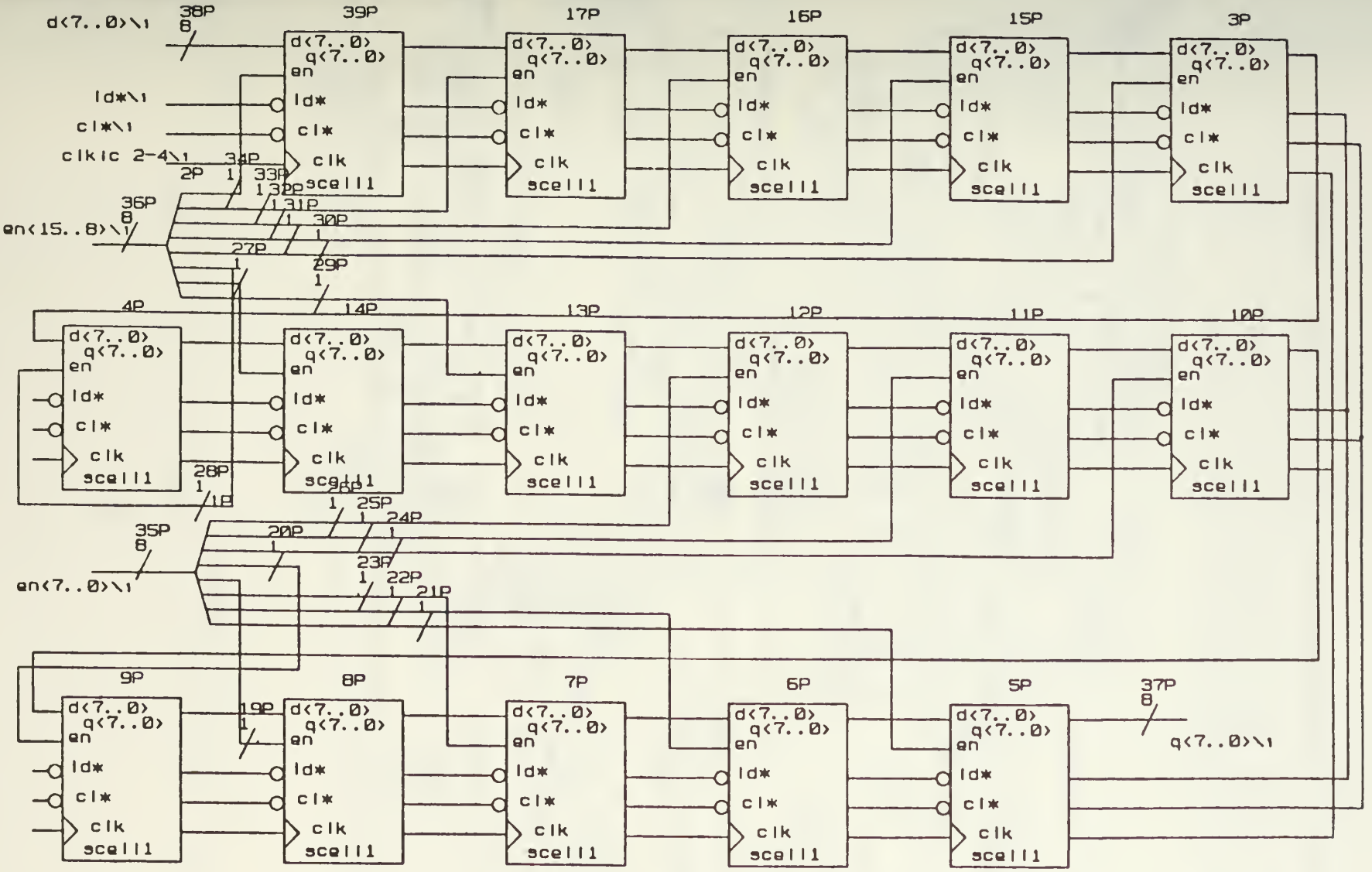


Fig. 5. The basic counter cell.

Using scell 1, a sixteen binary counter bank called scell 16 is constructed as shown in Fig. 6. There are enable inputs EN 15..0 that allow the counters to be incremented selectively. There are also parallel load paths between counters in the bank. Using these parts, the shiftcount is constructed as shown in Fig. 7. The shiftcount has a finite state machine that does operations at the leading edge of the clock. It sequences to the next state at the following edge of the clock. The shiftcount incorporates two scell 16 units. At the end of each CAM operation, the shiftcount is started. The finite state machine in shiftcount does the following sequence of operations:

	At the leading edge	At the falling edge
state 0:	load in the match word for low bank cells from CAM	change to state 1
state 1:	load in the match word for high bank cells from CAM	change to state 2
state 2:	send the 32 bit enable input to the two scell 16	change to state 3
state 3:	activate the binary counter bank for parallel shift upward	change to state 0, and stay there until the next start.



DRAWING

TITLE=SCELL16
ABBREV=scell16

LAST_MODIFIED=Thu Aug 22 15:07:13 1985

Fig. 6. 16 binary counter bank

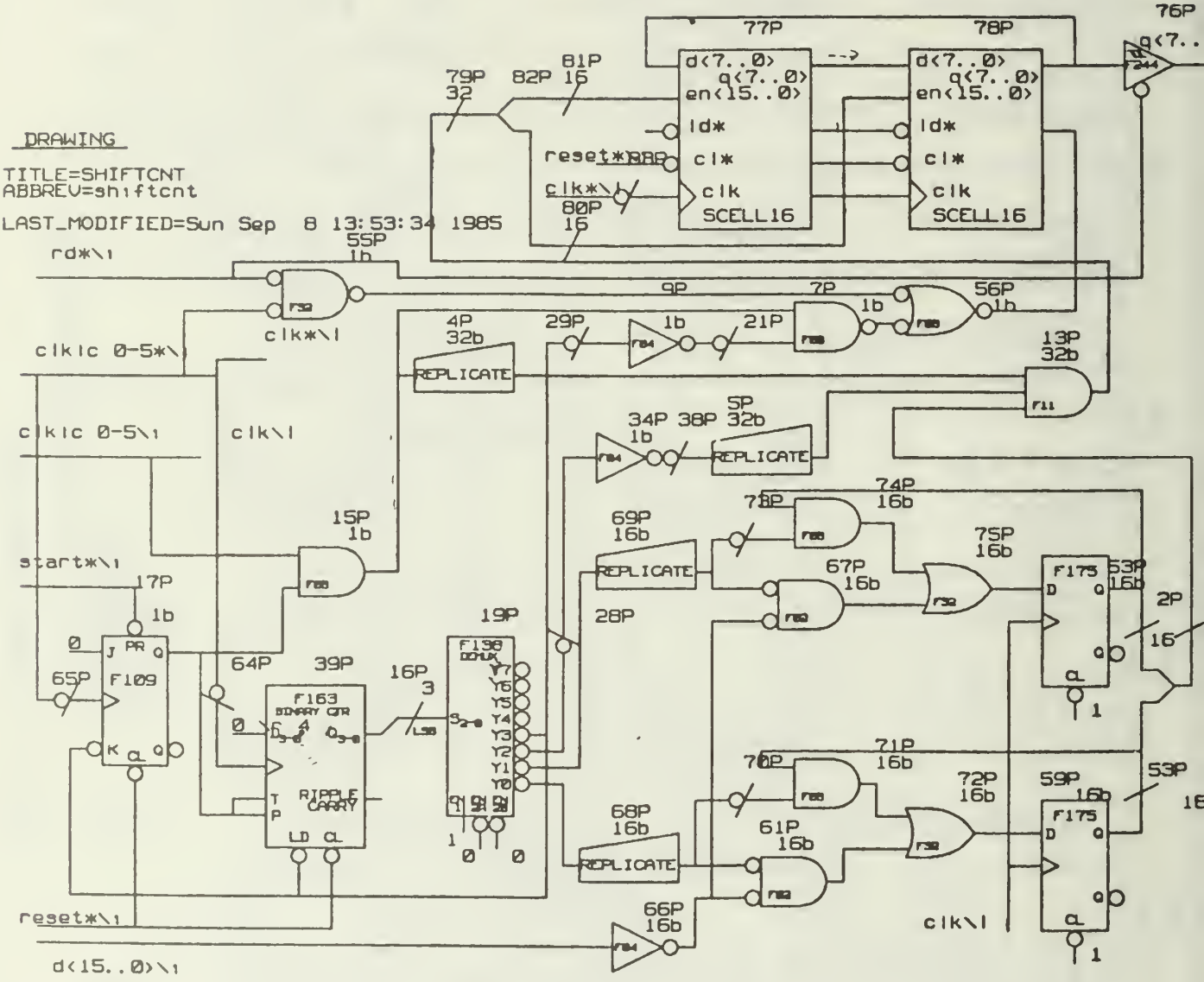


Fig. 7. Shift count Unit.

The interface between the shiftcount and the control unit is defined as

1. Reset: reset the finite state machine and the binary counter bank
2. Start: it starts the shiftcount operation after CAM output is available
3. CLK : clock input
4. Read : when this level is pulled low, the contents of the binary counter bank is shifted out at the trailing edge of the clock.

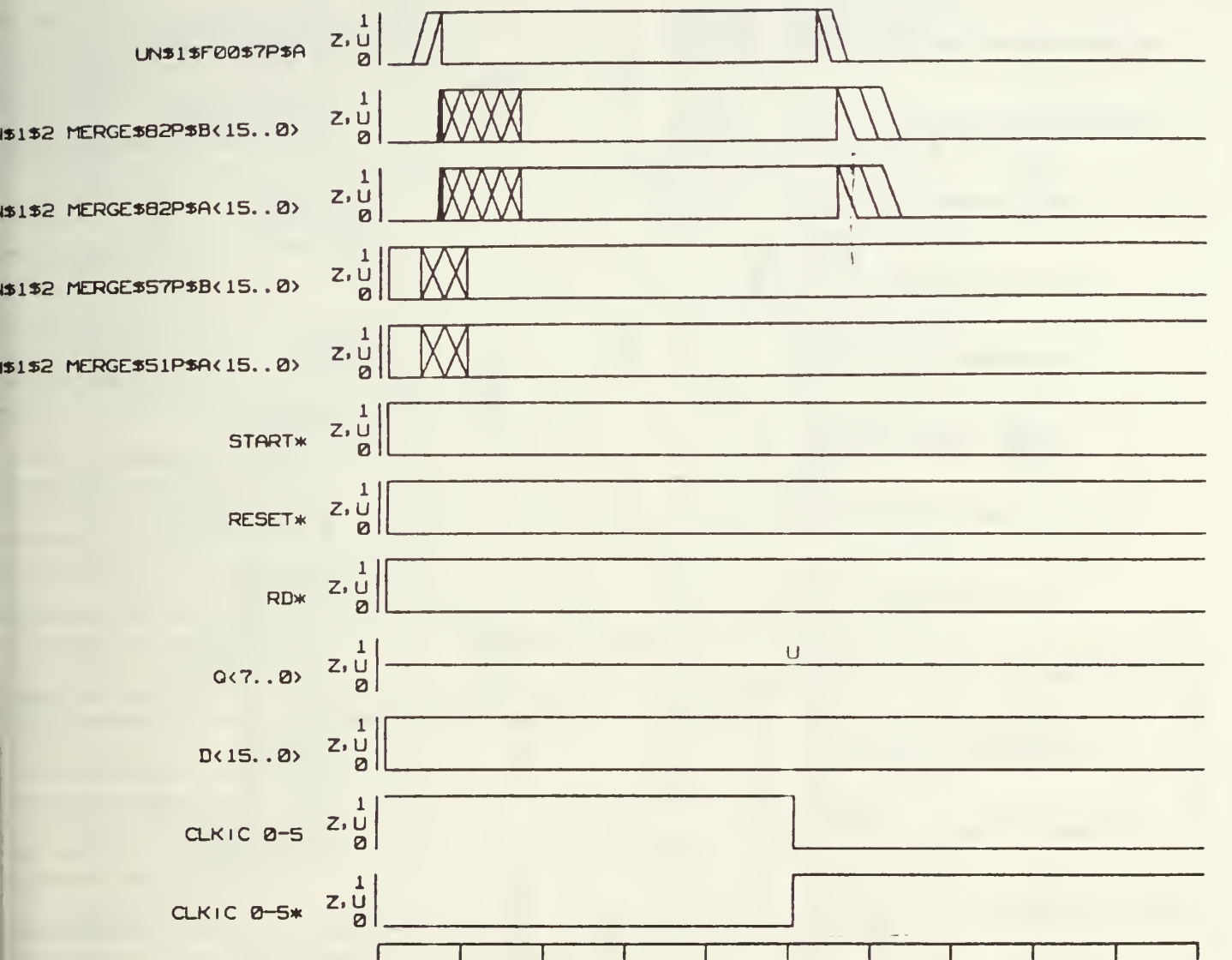
Timing Verification:

The binary counter bank of the two 8cell 16 units are triggered at the trailing edge of the clock. The timing verification results are shown in Fig. 8. This diagram shows the rising edge of clock at 0 nanoseconds and the trailing edge of the clock at 50 nanoseconds. For fixed stable START*, RESET*, RD* inputs, the change (unstable) states are shown as cross-hatch areas in the diagram. It is obvious that all the intermediate signals in the diagram as defined in the SCALD signal name sytax are stable at those two edges. Therefore, there are no timing problems in the proposed design. All the device propagation delays are included in the test. For the registers, the requirement on minimum pulse width, the setup, and hold time are also checked and verified.

Simulation:

The design has also been checked in the SCALD simulator. Part of the

waveform results is shown in Fig. 9. The start pulse synchronized with the clock is sent to the circuit. Different values of data input D [15..0] are provided to check whether the counter bank inscell 16's will work correctly. The reset* function is also checked to see whether the unit can be cleaned. The simulation was done from time 0 to time 6000 nanoseconds. The basic clock period is 100 nanoseconds so 120 possible events were tested in the simulation run. Whenever the RD* is pulled low as shown in Fig. 9, the stored counter results is output from the tri-state driver. The 19P$S <2..0> signal shows the states of finite state machine. As you can see that each time the start* is activated, the state machine will go through state 0, 1, 2, 3, and return back to zero. The 82P$Y <31..0> is the enable input to the counter bank that causes the selective counting. The signal of 56P$Y is the parallel load signal to the counter bank for shifting upward in Fig. 3. One problem that is part of the simulation process is its slow response time. For a moderate design like this it requires almost 50 minutes to get the results.




```
Timing Verification of Drawing 'SHIFTCNT';
( COMPILATION ON SUN SEP  8 16:08:56 1985   );
Verifier Directives:
CLOCK_PERIOD 100.0;
CLOCK_INTERVALS 10 (10.0ns);
CLOCK_SKEW 0.0;
```

Fig. 8. Timing Verification Results.

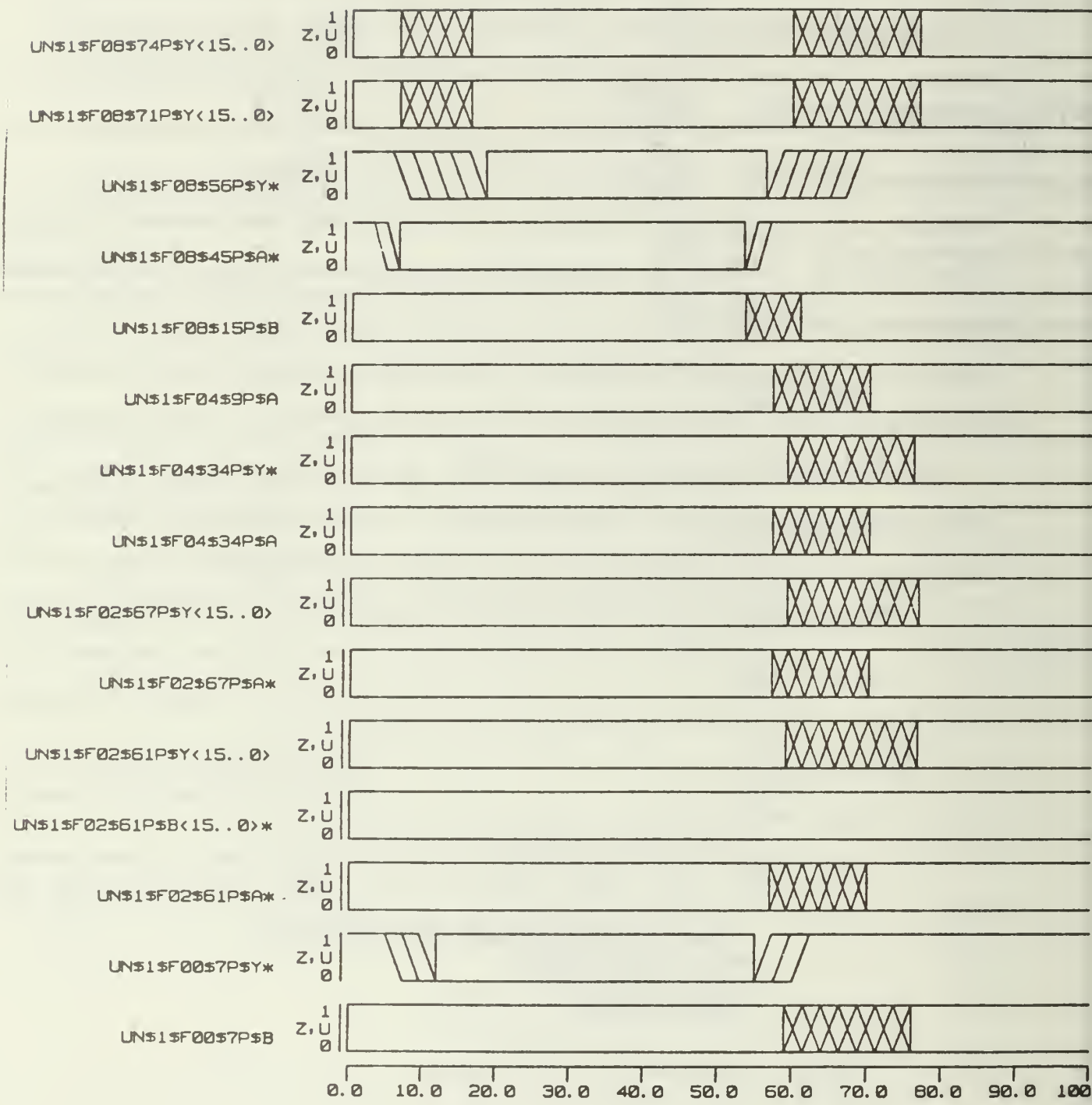


Fig. 8. (Continued)

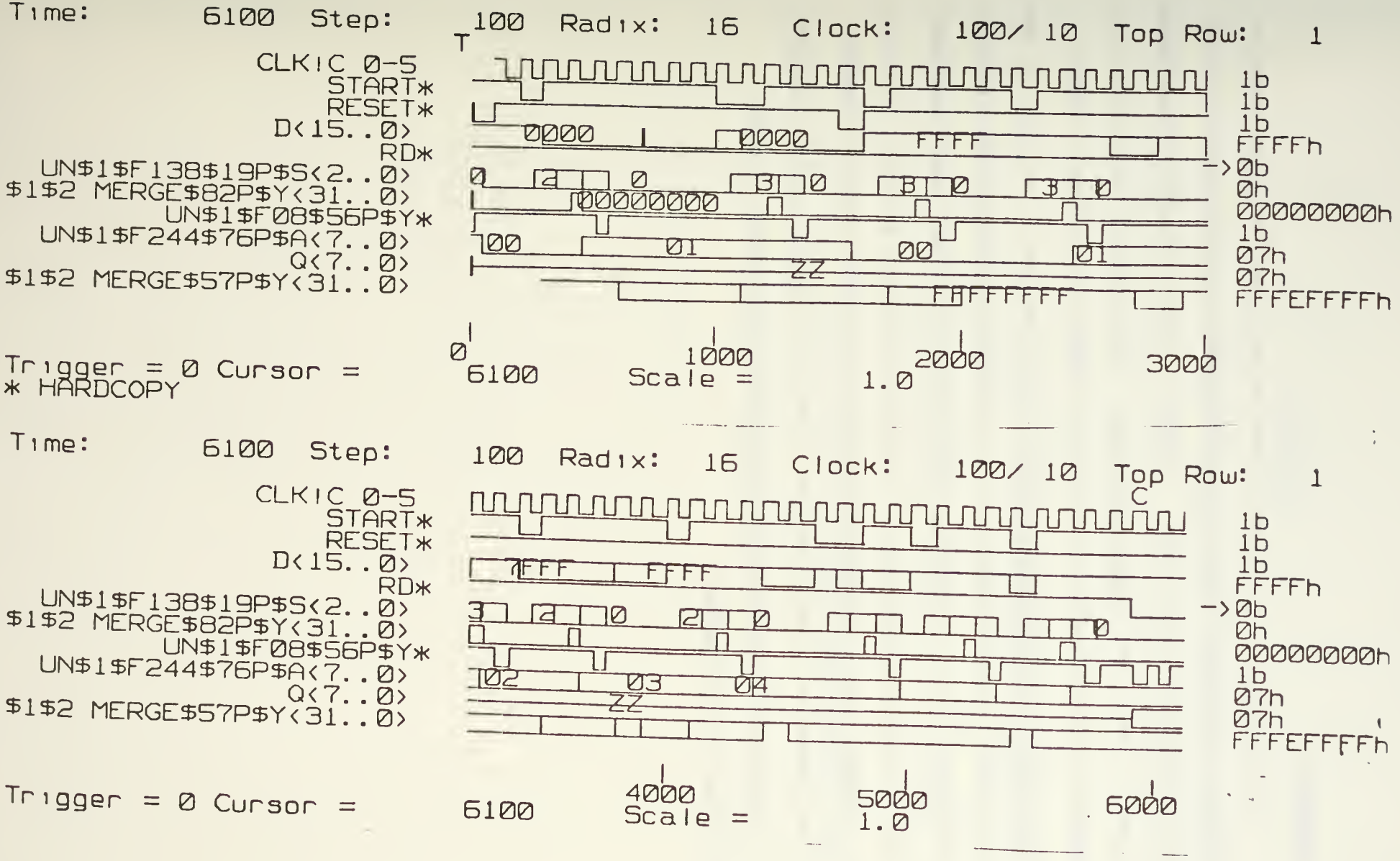


Fig. 9. Simulation Results.

Discussion:

In this paper, a new similarity counting algorithm for object detection in signals is studied. There are the conventional sequential algorithms and the presented parallel algorithm. A design is reported here that uses content addressable memory and large counter banks. This design constructed in hardware can allow object detection using M comparison operations on CAM. Regular correlation methods require L*M total operations. Presently, all the logic design of the shiftcount unit, the template, and the control unit for a one-dimensional case is completed. The details of the logic design together with the timing verification results and the simulation results are discussed here. Further study is underway to extend the controller design to include the two dimensional signal situation. Along with this study, the evaluation of a sequential similarity counting algorithm compared with the other algorithms is also underway.

References

(1) D.I. Barnea and H.F. Silverman, "A Class Of Algorithm For Fast Digital Image Registration"
IEEE Trans. Comp. vol. C-21, p.p. 179-186, Feb. 1972.

(2) M. Onoe and M. Saito, "Automatic Threshold Setting For The Sequential Similarity Detection Algorithm"
IEEE Trans. Comp. p.p. 1052-1053, Oct 1976.

(3) C-H Lee "Similarity Counting Algorithm For Image Registrations". to be submitted for publication.

Initial Distribution List

1.	Library, Code 1422 Naval Postgraduate School Monterey, CA 93943	2
2.	Professor C.H. Lee, Code 62Le Electrical and Computer Engineering Dept. Naval Postgraduate School Monterey, CA 93943	50
3.	Defense Technical Information Center Attn: DDC-TCA Cameron Station, Building 5 Alexandria, VA 22234	2
4.	C.E. Holland, Jr. Code 811 Naval Ocean System Center San Diego, CA 92152-5122	2
5.	Research Administration Office Code 012 Naval Postgraduate School Monterey, CA 93943	1